TRANSMITTAL FORM	Application Number	10/815,294
	Filing Date	Apr 1, 2004
	First Named Inventor	Barry, Edwin Franklin
	Art Unit	2188
(to be used for all correspondence after initial filing)	Examiner Name	Doan, Duc T.
Total Number of Pages in This Submission	Attorney Docket Number	800.0015 A01455
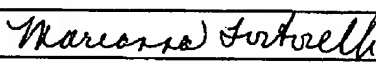
**RECEIVED
CENTRAL FAX CENTER
JUN 1 6 2008**

ENCLOSURES   *(Check all that apply)*

- [ ] Fee Transmittal Form
  - [ ] Fee Attached
- [ ] Amendment / Reply
  - [ ] After Final
  - [ ] Affidavits/declaration(s)
- [ ] Extension of Time Request
- [ ] Express Abandonment Request
- [ ] Information Disclosure Statement
- [ ] Certified Copy of Priority Document(s)
- [ ] Reply to Missing Parts/ Incomplete Application
  - [ ] Reply to Missing Parts under 37 CFR 1.52 or 1.53

- [ ] Drawing(s)
- [ ] Licensing-related Papers
- [ ] Petition
- [ ] Petition to Convert to a Provisional Application
- [ ] Power of Attorney, Revocation Change of Correspondence Address
- [ ] Terminal Disclaimer
- [ ] Request for Refund
- [ ] CD, Number of CD(s) _____
  - [ ] Landscape Table on CD

Remarks

- [ ] After Allowance communication to (TC)
- [ ] Appeal Communication to Board of Appeals and Interferences
- [x] Appeal Communication to TC (Appeal Notice, Brief, Reply Brief)
- [ ] Proprietary Information
- [ ] Status Letter
- [x] Other Enclosure(s) (please identify below):

PTO 2038 Credit Card Form

SIGNATURE OF APPLICANT, ATTORNEY, OR AGENT

Firm Name	Priest & Goldstein, PLLC		
Signature	*[signature]*		
Printed name	Peter H. Priest		
Date	June 16, 2008	Reg. No.	30210

CERTIFICATE OF TRANSMISSION/MAILING

I hereby certify that this correspondence is being facsimile transmitted to the USPTO or deposited with the United States Postal Service with sufficient postage as first class mail in an envelope addressed to: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450 on the date shown below.

Signature	*[signature]* Marianna Tortorelli		
Typed or printed name	Marianna Tortorelli	Date	June 16, 2008

RECEIVED
CENTRAL FAX CENTER

JUN 1 6 2008

800.0015   PATENT
A01455

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE
BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

In re Application of	:	Barry et al.
For	:	Methods and Apparatus for Address Translation Functions
Serial No.	:	10/815,294
Filed	:	04/01/2004
Group	:	2188
Examiner	:	Doan, Duc T.

Durham, North Carolina
June 16, 2008

MAIL STOP APPEAL BRIEF – PATENTS
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

TRANSMITTAL OF APPELLANT'S BRIEF

Dear Sirs:

1. Transmitted herewith is the APPEAL BRIEF in this application with respect to the Notice of Appeal filed on April 16, 2008.

2. The Applicant is other than a small entity.

3. Pursuant to 37 CFR 1.17(f) the fee for filing the Appeal Brief is $510.00.

   [x]   The Commissioner is hereby authorized to charge the fee of $510 our credit card.

   [   ]   The Commissioner is hereby authorized to charge the 1 month extension fee of $120 to our credit card. This letter petitions for a one month extension of time.

1

[X]   The Commissioner is hereby authorized to charge any additional fees which may
        be required or credit any overpayment to Law Offices of Peter H. Priest Deposit
        Account No. 50-1058.

                                            Respectfully submitted,

                                            Peter H. Priest
                                            Reg. No. 30,210
                                            Priest & Goldstein, PLLC
                                            5015 Southpark Drive, Suite 230
                                            Durham, NC 27713
                                            (919) 806-1600

2

800.0015
A01455

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE
BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

In re Application of	:	Barry et al.
For	:	Methods and Apparatus for Address Translation Functions
Serial No.	:	10/815,294
Filed	:	04/01/2004
Group	:	2188
Examiner	:	Doan, Duc T.

Durham, North Carolina
June 12, 2008

MAIL STOP APPEAL BRIEF – PATENTS
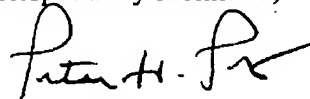Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

APPELLANTS' BRIEF

Sir:

1. The Real Party In Interest

    The real party in interest is the assignee, Altera Corporation.

2. Related Appeals and Interferences

    None.

06/17/2008 HMARZI1   00000013 10815294
01 FC:1402   510.00 OP

1

3.   Status of the Claims

This is an appeal from the December 260, 2007 final rejection of claims 1-19, all of the

pending claims. Claims 1, 2, 4-6, 16, and 17 were rejected under 35 U.S.C. § 103(a) based on

Dowling U.S. Patent No. 6,823,505 (Dowling) in view of Intel Pentium Processor Family

Developer's Manual Vol. 3 (Intel). Claims 3 and 19 were rejected under 35 U.S.C. § 103(a) as

unpatentable over Dowling, Intel as applied to claims 2 and 16, and in view of Nair et al. U.S.

Patent No. 6,944,747 (Nair). Claim 18 was rejected under 35 U.S.C. § 103(a) as unpatentable

over Dowling in view of Nair. Claims 7-10 were rejected under 35 U.S.C. § 103(a) as

unpatentable over Pechanek et al. U.S. Patent No. 6,173,389 (Pechanek 6,173,389) in view of

Dowling. Claims 11-14 were rejected under 35 U.S.C. § 103(a) as unpatentable over Dowling in

view of Choquette et al. U.S. Patent Publication No. 2002/0199084 (Choquette). Claim 15 was

rejected under 35 U.S.C. § 103(a) as unpatentable over Dowling, Choquette as applied to claim

14 and further in view of Nair. Pending claims 1-19 are the subject of this appeal.

4.   Status of Amendments

The claims stand as last amended on October 15, 2007. No Amendment After-Final has

been filed.

5.   Summary of Claimed Subject Matter

The present invention relates to address translation apparatuses and methods for

translating an address of a data element to a different address according to a translation pattern or

a function. Figs. 1-10 and the text at page 5 et seq. provide a detailed description of the

invention.

2

Claim 1

More particularly, claim 1 addresses a "processor address translation apparatus for translating an instruction operand address to a different operand address." As discussed at page 5, lines 16-18, a processor 100 shown in Fig. 1 may be adapted for use in conjunction with various embodiments of the present invention. Fig. 2D shows a processor subsystem 230 having an Rx read port translator 232 for translating instruction operand addresses to different addresses for reading addressable data from a register file 238, as described in detail at page 9, line 15 – page 10, line 15. As shown in Fig. 2D, an operand address bit field, provided from an instruction in the instruction register 231 on an Rx address bus 236, for example, is translated according to translation parameters stored in the translation parameter control unit 246 in response to the instruction received in the instruction register 231. See, for example, page 9, lines 15-23. The instruction having control information provided on a control input 252 from a decode and control unit, such as decode and control unit 118 in Fig. 1 as described at page 9, lines 19-21 and page 10, lines 6-15.

The apparatus of claim 1 comprises "a memory with an address input for selecting a data element from a plurality of data elements." The processor subsystem 230 of Fig. 2D shows a memory as the register file 238 having an address port 240 input that is supplied by the output of a port address register 239. A 1 to 8 selector 241 is used to decode the binary input provided on the address port 240 into one of eight selection signals 242 for selecting a data element from a plurality of data elements R0-R7 as described at page 10, lines 1-3.

The apparatus of claim 1 also comprises "an instruction register for receiving an instruction encoded with an operand address in an operand address bit field of the instruction and

3

control information indicating the operand address is to be translated as part of the instruction's execution." Instruction register (IR) 108 of Fig. 1 receives instructions from an instruction bus 106 that is coupled to a short instruction word memory 104. As an example, an arithmetic instruction is received in the IR 108 having three register file address fields, Rt 142, Rx 144, and Ry 146. The arithmetic instruction received in the IR 108 further supplies opcode and control bits 120 over an opA bus 154 to a decode and control unit 118 as described at page 6, lines 3-11 and at page 7, lines 11-22.

The apparatus of claim 1 further comprises "an address translation unit for accessing the memory in a translation pattern, having the operand address bit field as input and, in response to the instruction received in the instruction register, directly translating the operand address bit field received as input to form the different operand address in accordance with the translation pattern, the different operand address accessing a data element from the memory through the address input." In greater detail, the processor subsystem 230 of Fig. 2D shows an instruction register (IR) 231 that supplies an operand address bus 236 to an address translator 232. The address translator 232 also receives a control input 252 from a decode and control unit, such as decode and control unit 118 of Fig. 1, and a load translation parameter input 248 to specify the operation of the address translator 232. The translator 232 generates translated outputs A0', A1', and A2' 234 which are latched at the end of a decode stage in port address register 239. A number of different translation patterns are supported through use of a control input 244 which selects a particular translation operation. The values placed on the control input 244 may be based on a bit field in the instruction stored in the instruction register 231, such as an opcode or specific control bits as described at page 9, line 15 – page 10, line 15.

4

Claim 3

Claim 3 addresses the "processor address translation apparatus of claim 2". As described at page 13, lines 1-14, a general equation (3) that may be adapted for a k-bit address as described in further detail for use in "each address translator, such as address translators 139, 153, and 151 of Fig. 1" and used in conjunction with various embodiments of the present invention. Fig. 1 shows, for example, address translators 139, 151, and 153 in operand address paths for direct operand addressing instructions as described at page 5, line 16 – page 8, line 3.

As claimed in claim 3 the "translation parameters include k by k s bits and k e bits for a k bit address and address translation functions further comprises combinatorial logic governed by the following equations, where the operand address bit field input is A0, A1, ..., A(k-1), product operations are treated as ANDs, sum operations are treated as XORs, and the different operand address is A0', A1', ..., A(k-1)',

$$
\begin{pmatrix} A0' \\ A1' \\ \vdots \\ A(k-1)' \end{pmatrix} = \begin{pmatrix} s0 & s1 & \cdots & s(k-1) & e0 \\ sk & s(k+1) & \cdots & s(2k-1) & e1 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ s(k-1)k & s(k-1)k+1 & \cdots & s(k-1)(k+1) & e(k-1) \end{pmatrix} \times \begin{pmatrix} A0 \\ A1 \\ \vdots \\ A(k-1) \\ 1 \end{pmatrix}.
$$

." An example for a 32 entry register file, such as the register file 110 of Fig. 1, using a k=5-bit address requires for each translation matrix "25-bits of storage for the 5x5 s-bits and 5 bits of storage for the 5 e-bits" for five address lines A0-A4. Another example is described with three address lines A0'-A2' of equation 1 at page 11, line 22 – page 12, line 11, where "product operations are treated as ANDs, and sum operations are treated as XORs".

Claim 4

Claim 4 addresses the "processor address translation apparatus of claim 1 wherein the instruction is a block load instruction". As described at page 11, lines 3-16 with regard to Fig. 3,

5

a block load instruction is described for use with address translation. The block size to be loaded

is encoded in block size field 315 and a translation option is encoded in a two bit translation

selection (Tsel) field 305. For example, the block load instruction may be used "to load a linear

sequential ordering of data from a memory into a bit-reverse address pattern in a register file" as

described at page 11, lines 5-8.

Claim 7

Claim 7 addresses a "processor register file indexing (RFI) address translation apparatus

for translating an RFI sequence of instruction operand addresses to an RFI sequence of different

operand addresses." As described at page 13, line 15 – page 14, line 4, an exemplary RFI VLIW

processor 400 may be adapted as described in further detail in the discussions of Figs. 5 and 6 for

use in conjunction with various embodiments of the present invention. Fig. 5, for example,

shows further details of an ALU subsystem 500 of the RFI VLIW processor 400 of Fig. 4 with

address translation functions in each operand address for translating RFI addresses, as described

at page 15, line 22 – page 16, line 23. Fig. 6, for example, shows further details of address

translation operation in RFI addressing in a data select unit (DSU) subsystem 600 of the RFI

VLIW processor 400 of Fig. 4 as described at page 17, line 1 – page 19, line 3.

The apparatus of claim 7 comprises "a memory with an address input for selecting a data

element from a plurality of data elements." The ALU subsystem 500 of Fig. 5 shows a memory

as the register file 536 having address ports 544, 546, and 548 which are used to access operands

from a block of operands stored in the register file 536 as described at page 16, lines 19-23. The

DSU subsystem 600 of Fig. 6 described further details of the RFI address translation apparatus.

As described at page 18, lines 15-17, register file 626 is a memory which receives operand

addresses on Rt address port 624 for use in accessing operands from the register file 626.

6

The apparatus of claim 7 comprises "an instruction register for receiving an instruction encoded with an operand address and control information indicating the operand address is to be translated as part of the instruction's execution." The DSU subsystem 600 of Fig. 6 shows an instruction register (IR) as DSU slot IR 602 receiving a DSU instruction from a VLIW DSU slot instruction bus which is one of the slot instruction buses from VLIW bus 434 of the RFI VLIW processor 400 of Fig. 4. As an example, the DSU slot IR 602 contains an instruction encoded with multiple fields such as Rt(5) field 614, a PEXCHG opcode, opcode extensions, and a Tsel field 642. The instruction is decoded in a pipeline decode stage 412 providing control information for the execution of the received instruction in a similar manner to the description of address translation in the processor 100 of Fig. 1 at page 7, lines 21-22, as indicated in the ALU subsystem 500 of Fig. 5 at page 16, lines 22-23, and as indicated in the DSU subsystem 600 of Fig. 6 at page 18, line 22 – page 19, line 3.

The apparatus of claim 7 comprises "an RFI update unit enabled to generate on the RFI update unit's output a linear sequence of RFI operand addresses in response to a received sequence of RFI translation type instructions." The DSU subsystem 600 of Fig. 6 shows an RFI update unit 610 which responds to an RFI enable signal and RFI parameter control information as described in further detail in Figs. 4, 5, and 6. Beginning with Fig. 4, the RFI VLIW processor 400 describes a VLIW memory (VIM) controller 430 which determines an RFI enable signal during a predecode stage 408 for each of the RFI enabled XV instructions that access the same VIM 432 location as described at page 14, line 16 – page 15, line 2. Each time an RFI enabled XV instruction having the same VIM location is decoded, the RFI operation is enabled. The same VIM location indicates the same VLIW slot instructions, such as the ALU instruction received in the ALU slot IR 534 and the DSU instruction received in the DSU slot IR 602, are to

7

be executed. For example, the ALU decode, RFI and translator control unit 530 responds to an

RFI enable signal 532 generated in a VMC, such as VMC 430, in response to received execute

VLIW (XV) instructions containing RFI control information as described at page 16, lines 7-19.

Each execution of the same VLIW slot instruction causes the operand address to be updated via

the RFI update unit 610 generating a linear sequence of operand addresses in response to the

received sequence of RFI translation type instructions as described at page 17, lines 7-17 and at

page 18, lines 17-19.

The apparatus of claim 7 also comprises "a multiplexer for selecting between the operand

address from the instruction register for a first RFI operation and selecting the RFI update unit's

output for subsequent RFI operations." Multiplexer 616 of DSU subsystem 600 of Fig. 6 is used

to select on a first RFI operation the operand address 614 from the DSU slot IR 602 and on

subsequent RFI operations to select updated operand addresses from a look ahead register 620

located in the RFI update unit 610 to generate a sequence of RFI operand addresses as described

at page 17, line 17 – page 18, line 6.

The apparatus of claim 7 further comprises "an address translation unit for accessing the

memory in a translation pattern, receiving a sequence of operand addresses from the multiplexer

and, in response to the sequence of RFI operand addresses, translating the sequence of RFI

operand addresses to form a sequence of different operand addresses in accordance with the

translation pattern, the different operand addresses each accessing a data element from the

memory through the address input." Rt address translator 634 of the DSU subsystem 600 of Fig.

6 receives the sequence of RFI operand addresses from the multiplexer 616 on output bus 630

and based on the translation type instruction translates the sequential stream of addresses to a

desired address sequence on the translator bus 636. The translator bus 636 is latched in the port

8

address register 618 at the end of each decode stage 412 for each RFI instruction executed and

the different operand addresses are provided on the Rt address port 624 each accessing a data

element of the register file 626. The translation pattern, as controlled by a specific set of {s,e}

bits, is selected by use of the Tsel bit field 642. See, for example, the description at page 18, line

9 – page 19, line 3. The translation pattern may be determined from equations (1) and (2) based

on the {s,e} bits as described at page 11, line 17 – page 12, line 11.

Claim 11

Claim 11 addresses an "address translation memory device for accessing data at

translated addresses." As discussed at page 21, line 10 – page 22, line 3, a storage subsystem

800 shown in Fig. 8A includes a storage unit 810 which receives address inputs 815 and outputs

data on read port output Rx 825 accessed at a translation of the address inputs 815.

The address translation memory device of claim 11 comprises a "first read address input

port to the address translation memory device." The address inputs 815 to the "storage unit 810,

showing only a read port path" comprise the first read address input, as described at page 22,

lines 4-5 and lines 8-10.

The address translation memory device of claim 11 also comprises a "storage device

located in the address translation memory device having data accessible at addressable locations,

a second read address input port internal to the address translation memory device for selecting

data from the storage device during read operations, and a data output port." The storage

subsystem 800 of Fig. 8A shows a storage device 835 which includes a read address input port to

a location selector, such as 1 of 8 selector 840 having an address input A2' A1' A0', a storage

9

array 845 with accessible locations R0-R7, and data output port Rx 825 as described at page 22, lines 17-19.

The address translation memory device of claim 11 further comprises "an address translation unit located in the address translation memory device for accessing the storage device in a translation pattern, the address translation unit translating a first read address value coupled to the first read address input port in accordance with the translation pattern, to a second read address value coupled to the storage device second read address input port for reading data from the storage device at a translated address during a read operation." The storage subsystem 800 of Fig. 8A shows an address translator 830 which applies {s,e} information to translate the address input A2 A1 A0 to the address input A2' A1' A0' prior to accessing the storage device 835. The translation pattern is controlled by use of equation (2) for example using the {s,e} information as described at page 11, line 17 – page 12, line 11 and at page 22, lines 8-17.

Claim 15

Claim 15 addresses the "address translation memory device of claim 14". As described at page 13, lines 1-14, a general equation (3) that may be adapted for a k-bit address as described in further detail for use in "each address translator, such as address translators 139, 153, and 151 of Fig. 1" and used in conjunction with various embodiments of the present invention. Fig. 1 shows, for example, address translators 139, 151, and 153 in operand address paths for direct operand addressing instructions as described at page 5, line 16 – page 8, line 3. Fig. 8A also shows an address translator 830 located in the address translation memory device which "applies {s,e} bit state information to translate the address input for any access to the storage device 835" as described at page 22, lines 4-17.

10

As claimed in claim 15 the "translation parameters include k by k s bits and k e bits for a

k bit address and address translation functions further comprises combinatorial logic governed by

the following equations, where the first read address input is A0, A1, ..., A(k-1), product

operations are treated as ANDs, sum operations are treated as XORs, and translated address

output are A0', A1', ..., A(k-1)',

$$
\begin{pmatrix} A0' \\ A1' \\ \vdots \\ A(k-1)' \end{pmatrix} = \begin{pmatrix} s0 & s1 & \cdots & s(k-1) & e0 \\ sk & s(k+1) & \cdots & s(2k-1) & e1 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ s(k-1)k & s(k-1)k+1 & \cdots & s(k-1)(k+1) & e(k-1) \end{pmatrix} \times \begin{pmatrix} A0 \\ A1 \\ \vdots \\ A(k-1) \\ 1 \end{pmatrix}.
$$
" An example for a

32 entry register file, such as the register file 110 of Fig. 1, using a k=5-bit address requires for

each translation matrix "25-bits of storage for the 5x5 s-bits and 5 bits of storage for the 5 e-bits"

for five address lines A0-A4. Another example is described with three address lines A0'-A2' of

equation 1 at page 11, line 22 – page 12, line 11, where "product operations are treated as ANDs,

and sum operations are treated as XORs".

Claim 16

Claim 16 addresses a "processor address translation method for translating an instruction

operand address to a different operand address." As discussed at page 5, lines 16-18, a processor

100 shown in Fig. 1 may be adapted for use in conjunction with various embodiments of the

present invention. Fig. 2D shows a processor subsystem 230 having an Rx read port translator

232 for translating instruction operand addresses to different addresses for reading addressable

data from a register file 238 as described in detail at page 9, line 15 – page 10, line 15.

The method of claim 16 comprises "receiving an instruction encoded with an operand

address in an operand address bit field of the instruction and control information indicating the

operand address is to be translated as part of the instruction's execution." Instruction register

11

(IR) 108 of processor 100 of Fig. 1 receives instructions from an instruction bus 106 that is coupled to a short instruction word memory 104. As an example, an arithmetic instruction is received in the IR 108 having three register file address fields, Rt 142, Rx 144, and Ry 146. The arithmetic instruction received in the IR 108 further supplies opcode and control bits 120 over an opA bus 154 to a decode and control unit 118 as described at page 6, lines 3-11 and at page 7, lines 11-22.

The method of claim 16 also comprises "translating directly the operand address bit field received as input according to a function." A block load with address translation instruction 300 of Fig. 3 represents an exemplary function. The block load instruction is encoded with a two bit translation selection (Tsel) field 305 that is used to select a translation pattern. A number of different translation patterns are supported through use of a control input 244 in Fig. 2D which selects a particular translation operation. The values placed on the control input 244 are based on a bit field in the instruction stored in the instruction register 231, such as the Tsel field 305 bits, as described at page 11, lines 3-16. The translation pattern, as controlled by a specific set of {s,e} bits, may be determined from the functions represented by equations (1) and (2) based on the {s,e} bits as described at page 11, line 17 – page 12, line 11 and at page 13, lines 12-14.

The method of claim 16 also comprises "accessing a data element with the translated address, and repeating the receiving, translating, and accessing steps to access data elements in a pattern according to the function." The block load with address translation instruction 300 of Fig. 3 may specify a block of data to be loaded from a linear sequence of data located in a local data memory to locations in a register file using bit-reverse operand addresses according to the Tsel field 305 as described at page 11, lines 3-17. For each data element in the block of data, the

12

steps of receiving, translating, and accessing are repeated until the block size, as specified by the block size field 315, is reached.

Claim 18

Claim 18 addresses an "address translation method for translating a first address of a first data element in an address translation memory to a second address of a second data element in the address translation memory." See, for example, Figs. 8A, 9, 10A, and 10B, page 21, line 10 – page 22, line 1, and page 24, line 10 – page 26, line 5 which describe translation operations including a transpose operation.

The method of claim 18 comprises "determining a set of {s, e} bits that specify a translation pattern." An {s,e} matrix 6 is determined for a transpose operation as described at page 25, line 14 – page 26, line 5. Other translation patterns may be specified by determining a different {s,e} matrix.

The method of claim 18 comprises "loading the set of {s, e} bits into an address translation parameter control register located in the address translation memory." A translation parameter control unit 870 of Fig. 8B is located in an address translation memory storage unit 855 as describe at page 23, lines 1-11. Translation parameters, such as {s, e} bits are loaded and held in the translation parameter control unit. The two port memory unit 900 of Fig. 9 shows a load translation pattern input 935 that is used to load the {s,e} bit translation state information as described at page 24, lines 13-16 and page 25, lines 16-17.

The method of claim 18 comprises "enabling an address translation unit located in the address translation memory for translation." Rx address translator 830 of storage subsystem 800 is enabled by the loading of translation parameters 820 which govern how the addresses access data from internal storage as described at page 21, line 22 – page 22, line 1. In a similar manner,

13

the {s,e} matrix 6 is loaded into a parameter control register that is part of translator 940 in Fig. 9

to govern how data is to be accessed for the transpose operation as described at page 25, lines

16-18.

The method of claim 18 comprises "initiating a read operation to read a first data element

at a first address during a read operation." As described at page 25, lines 18-20 in reference to

Figs. 9 and 10B, when data is read the address may be supplied in a linear order as indicated in

the left column as binary five bit addresses in table 1028.

The method of claim 18 also comprises "translating the first address to the second

address in accordance with the {s, e} bit specified translation pattern." Equation (4) at page 19,

lines 20-23 as specified with the {s,e} matrix 6 values is applied to the read address received at

the read port address 925. This application of equation 4 translates the received address

according to the specified transpose pattern of the {s,e} matrix 6 to a translated address.

The method of claim 18 further comprises "completing the read operation by reading the

second data element at the second address." As shown in Fig. 10B and described at page 25,

lines 14-20, an input read address, as specified from the left column of table 1028, is translated

and data is then read as specified in the right column of table 1028 using matrix notation which is

further illustrated in matrix 5 1024.

6.   Grounds of Rejection to be Reviewed on Appeal

Claims 1, 2, 4-6, 16, and 17 were rejected under 35 U.S.C. § 103(a) based on Dowling in

view of Intel. Claims 3 and 19 were rejected under 35 U.S.C. § 103(a) as unpatentable over

Dowling, Intel as applied to claims 2 and 16, and in view of Nair. Claim 18 was rejected under

35 U.S.C. § 103(a) as unpatentable over Dowling in view of Nair. Claims 7-10 were rejected

under 35 U.S.C. § 103(a) as unpatentable over Pechanek 6,173,389 in view of Dowling. Claims

14

11-14 were rejected under 35 U.S.C. § 103(a) as unpatentable over Dowling in view of

Choquette. Claim 15 was rejected under 35 U.S.C. § 103(a) as unpatentable over Dowling,

Choquette as applied to claim 14 and further in view of Nair.

7.   Argument

The final rejections under 35 U.S.C. § 103 did not follow M.P.E.P. § 706.02(j) which

states:

> After indicatin g that the rejection is under 35 U.S.C. 103, the Examiner should set
> forth...the difference or differences in the claim over the applied
> reference(s),...the proposed modification of the applied reference(s) necessary to
> arrive at the claimed subject matter, and ... an explanation why one of ordinary
> skill in the art at the time the invention was made would have been motivated to
> make the proposed modification.

As will be illustrated below, the claims of the present invention are not obvious in view of the

references relied upon by the Examiner.

A.   Rejections under Section 103(a)

These rejections are not supported by the relied upon art. 35 U.S.C. § 103 which governs

obviousness indicates that "differences between the subject matter sought to be patented and the

prior art" are to be assessed based upon "the subject matter as a whole". Analyzing the entirety

of each claim, the rejections under 35 U.S.C. § 103 are not supported by the relied upon art as

addressed further below. Only after an analysis of the individual references has been made can it

then be considered whether it is fair to combine teachings. However, as addressed further below,

fairness requires an analysis of failure of others, the lack of recognition of the problem, and must

avoid the improper hindsight reconstruction of the present invention. Such an analysis should

consider whether the modifications are actually suggested by the references rather than assuming

15

they are obvious. The 35 U.S.C. § 103 rejections made here pick and choose elements from two or more separate references, neither of which presents any motivation for making the suggested combination. This approach constitutes impermissible hindsight and must be avoided. As required by 35 U.S.C. § 103, claims must be considered as a whole. When so considered, the present claims are not obvious.

Claims 1, 2, 4-6, 16, and 17

The Official Action rejected claims 1, 2, 4-6, 16, and 17 under 35 U.S.C. §103(a) as being anticipated by Dowling in view of Intel. Dowling addresses programmable address modes for calculating effective addresses based on address values stored in address registers for operands stored in a data memory. Dowling, Abstract. Dowling's processor fetches a data element from a data memory 120 in response to an address generated in an address arithmetic unit, such as AAU 106 or programmable AAU 212 of Fig. 2, based on one of the address registers AR0, AR2, ... ARn of register set 102. Dowling, col. 7, lines 48-50 and col. 8, lines 35-48, for example. Dowling describes such effective address generation by a programmable AAU in more detail with regard to Figs. 3A and 3C. In Figs. 3A and 3C, it is shown that the programmable AAUs 301 and 350, respectively, both receive an address value as input from the address registers. Dowling further states that the programmable AAU 301 may be used "as the programmable AAU 212 in the processor 200 shown in Fig. 2" which "allows a programmer to programmably permute the bits in the address registers AR0-ARn in the register set 102." Dowling, Figs. 2, 3A, and 3C and col. 10, lines 14-18. Dowling merely extends the capabilities of an address arithmetic unit (AAU) by programmably modifying address values stored in address registers.

In addition, the Examiner suggests that Dowling discloses an instruction register for

16

receiving an instruction encoded with an operand address and control information indicating the operand address is to be translated as part of the instruction's execution. The Examiner cites Dowling's table 5 and col. 13, lines 19-35 for support. The Examiner further suggests that in response to the instruction received in the instruction register, Dowling translates an operand address to form the different operand address. The Examiner cites the same table 5 and col. 13, lines 19-35 for support.

However, Dowling's table 5 lists program code made up of load and store instructions with an operand addresses that specify address registers, such as address register 0 (AR0), address register 4 (AR4), and address register 5 (AR5). The load and store instruction encoding would require a bit field to specify these address registers. **Dowling does not transform this bit field encoded with the selected address register, but rather operates to transform the contents of the selected address register, as noted above with regard to Dowling's Figs. 3A and 3C.**

The Examiner correctly admits that Dowling does not expressly disclose "directly translating the operand (address) bit field received as input..." when doing the address translation. The Examiner suggests that Intel discloses a direct addressing format and cites Intel page 25-6 disp16. However, at Intel page 25-6 Notes: 2 "disp16 denotes a 16-bit displacement following the ModR/M bytes, to be added to the index." As described in Intel, disp16 is not a direct addressing format but, rather, is used in the generation of an effective address by adding the disp16 value to an index value. Intel does not teach and does not make obvious an "instruction register for receiving an instruction encoded with an operand address in an operand address bit field of the instruction and control information indicating the operand address is to be translated as part of the instruction's execution" as claimed in claim 1.

17

The Examiner further states that it would have been obvious to one of ordinary skill in the art at the time of invention to include an operand direct addressing format as suggested by Intel in Dowling's system. However, Dowling does not show a direct addressing path from an instruction register for receiving an instruction with an operand address and control information as claimed in claim 1. It is also not clear how Dowling could provide such a path since Dowling does not mention an instruction register. Further, Dowling provides no motivation that such a path would provide any value. Rather, Dowling states that the programmable AAU 301 may be used "as the programmable AAU 212 in the processor 200 shown in Fig. 2" which "allows a programmer to programmably permute the bits in the address registers AR0-ARn in the register set 102." Dowling, Figs. 2, 3A, and 3C and col. 10, lines 14-18. Dowling merely extends the capabilities of an address arithmetic unit (AAU) by programmably modifying address values stored in address registers. Dowling does not teach and does not make obvious an "instruction register for receiving an instruction encoded with an operand address in an operand address bit field of the instruction and control information indicating the operand address is to be translated as part of the instruction's execution" as claimed in claim 1. Also, Dowling does not teach and does not make obvious "an address translation unit for accessing the memory in a translation pattern, having the operand address bit field as input and, in response to the instruction received in the instruction register, directly translating the operand address bit field received as input to form the different operand address in accordance with the translation pattern, the different operand address accessing a data element from the memory through the address input" as claimed in claim 1.

Regarding claim 4, the Examiner suggests that Dowling's Fig. 4A and column 10, lines 30-65 disclose a block load instruction. The Official Action further suggests that a block load

18

instruction is a typical matrix operation instruction in processor TMS320C2x. However,

Dowling's Fig. 4A merely illustrates a matrix having sixteen data elements. The text at

Dowling's col. 10, lines 30-60 does not indicate a block load instruction, but merely indicates

that the data elements of the matrix 400 shown in Fig. 4A are **"actually stored in a linear array**

in memory" and as "shown in FIG. 4B, in memory, the sixteen elements of the matrix 400 are

laid out sequentially in memory". Emphasis added. Dowling, col. 10, lines 58-60 and lines 34

and 35. Dowling makes no mention of a singular instruction which may load a block of data as

claimed. Dowling does not teach and does not make obvious a block load instruction as claimed.

Regarding claim 16, the Examiner relies upon Dowling and the suggested rationale used

in the rejection of claim 1 to reject claim 16. As noted above, the suggested rationale for

rejecting claim 1 is not valid. Hence, Dowling does not teach and does not make obvious claim

16.

Claims 3 and 19

Claims 3 and 19 were rejected under 35 U.S.C.§103(a) based on Dowling, Intel as

applied to claims 2 and 16, and in view of Nair. Nair addresses a matrix data processor and "an

indexing methodology 300" operating on data elements of various resolutions, such as 4-bit, 8-

bit, 16-bit, and so forth, by selecting the data elements by using indices representing the data

elements. Nair, col. 6, lines 29-48. Nair's indexing methodology 300 includes a starting index of

a first element of a first source matrix, a starting index for a first element of a second source

matrix, a starting index for a first element of a destination matrix, and an interval between

elements for selecting among successive elements of the matrices. Nair, col. 8, lines 47-50 and

col. 10, line 65 – col. 11, line 15.

19

Regarding claim 3, the Examiner admits that Dowling does not expressly disclose

providing the bit complementing function for matrix operation corresponding to the claim's "e

bits" function. The Examiner depends upon Nair to resolve this deficiency. The Examiner

suggests that Nair describes an apparatus for matrix processing that can be incorporated into

Dowling's system modified by Intel to manipulate address bits using matrix operations and cites

Nair col. 12, table 1 and col. 13, table 2 for support. Nair's, col. 12, table 1 and col. 13, table 2,

lists specific matrix operations that can be performed on data elements of a matrix. Nair, col. 12,

lines 14-22. The Examiner focuses on one particular aspect of claim 3 concerning the claim's

"e" bits function. The Examiner suggests that Nair teaches a bit complementing function that

may be applied to an operand address bit field. However, Nair merely describes operations on

data elements and not address bits. Also, Nair does not teach and does not make obvious

translation parameters that "include k by k s bits and k e bits for a k bit address" as claimed in

claim 3. Further, Nair does not teach and does not make obvious "combinatorial logic governed

by the following equations, where the operand address bit field input is A0, A1, ..., A(k-1),

product operations are treated as ANDs, sum operations are treated as XORs, and the different

operand address is A0', A1', ..., A(k-1)',

$$
\begin{pmatrix} A0' \\ A1' \\ \vdots \\ A(k-1)' \end{pmatrix} = \begin{pmatrix} s0 & s1 & \cdots & s(k-1) & e0 \\ sk & s(k+1) & \cdots & s(2k-1) & e1 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ s(k-1)k & s(k-1)k+1 & \cdots & s(k-1)(k+1) & e(k-1) \end{pmatrix} \times \begin{pmatrix} A0 \\ A1 \\ \vdots \\ A(k-1) \\ 1 \end{pmatrix}
$$
" as claimed in claim

3.

As noted above, Dowling does not teach and does not make obvious the processor

address translation apparatus of claim 1 and further provides no motivation to take data elements,

such as the data elements of Nair, and treat a data element as an "operand address bit field input".

20

Further, Dowling as modified by Intel does not teach and does not make obvious translation

parameters that "include k by k s bits and k e bits for a k bit address" as claimed in claim 3 and

Nair provides no recognition of claim 3's "k by k s bits and k e bits for a k bit address". Nair

does not teach and does not make obvious claim 3 and does not resolve the admitted deficiency

of Dowling.

Regarding claim 19, the Examiner relies upon the suggested rationale used in the

rejection of claim 3 to reject claim 19. As noted above, the suggested rationale for rejecting

claim 3 is not valid. Dowling as modified by Intel does not teach and does not make obvious

claim 19.

Claim 18

Claim 18 was rejected under 35 U.S.C. § 103(a) as unpatentable over Dowling in view of

Nair. Claim 18 addresses an "address translation method for translating a first address of a first

data element in an address translation memory to a second address of a second data element in

the address translation memory." Claim 18 claims "loading the set of {s,e} bits into an address

translation parameter control register located in the address translation memory" and "enabling

an address translation unit located in the address translation memory for translation." In contrast,

Dowling's data memory 120 has an address input, a bidirectional data port, and a data memory

storage. Dowling, Fig. 2 and col. 7, lines 56-59.

In the Official Action dated 12/26,2007 at page 17, the top paragraph, the Examiner

mistakenly quoted one of the Applicant's arguments with regard to claim 18 presented in the

Amendment dated 10/15,2007. In particular, the Examiner stated the Applicants argued that

"Dowling's data memory 102 is not an address translation memory". In the Amendment dated

10/15/2007 at page 14, lines 14 and 15, the Applicants' argued that "Dowling's data memory 120

21

is not an address translation memory". Dowling identifies a register set 102 as comprising address registers that contain address values and a separate data memory 120 that contains data values. The Examiner further suggests that "Dowling's data memory 102 have the data correspond to the address registers AR0 to ARn" in the Official Action dated 12/26/2007 at page 17, lines 3 and 4. Again, Dowling's data memory 120 is mistakenly identified with the reference label for the register set 102. Due to this error, it is not clear what the Examiner's is arguing. Yet it is due to this error that the Examiner dismisses the Applicants' remaining arguments as not being relevant. **Dowling's data memory 120 is not an "address translation memory" having an "address translation parameter control register" or having an "address translation unit" as claimed in claim 18.** Rather, Dowling's data memory 120 is a separate memory having no additional internal capabilities, that may be controlled by an "address translation parameter control register" or "an address translation unit" as claimed in claim 18. Dowling does not teach and does not make obvious an address translation method as claimed in claim 18.

The Examiner suggests that claim 18 merely rearranges some components of the address translation logic to be located in "the address translation memory" and such a rearrangement of old elements with each performing the same function it has been known to perform and yields no more than one would except from such an arrangement would be obvious. However, as noted in the specification at page 21, lines 10-15, the address translator is included in a register file or memory unit such as shown in Fig. 8A. In particular, "[c]consider an exemplary storage subsystem 800 of Fig. 8A, illustrating aspects of a read port, which operates differently than the storage subsystem 238 shown in Fig. 2D". Also, as noted in the specification at page 21, line 18 – page 22, line 1, in "Fig. 2D subsystem 230, the address translation occurs only in operation for instructions that specify the address translation function. Other instructions, which don't specify

22

an address translation function, use the register file or memory unit normally as sequentially addressed storage. In the storage subsystem 800 of Fig. 8A, all address inputs 815 are translated according to the translation settings 820 that govern how the addresses access data from the storage unit 835". Claim 18 does not claim a rearrangement of old elements providing an already known function.

The Examiner suggests that Dowling's Fig. 2 teaches each element of claim 18's "address translation memory". However, Dowling does not initiate a "read operation to read a first data element at a first address during a read operation" where the "first address of a first data element in an address translation memory" as claimed in claim 18. Dowling translates an address before any read operation to the data memory 120 occurs. As shown in Dowling's Fig. 3A, an input from address register 302 is permuted to an output to address register 304. As also shown in Dowling's Fig. 2, the output of programmable AAU 212 is selected by mux 203 and provided to the register set 102 comprising address registers. It is the output of the address registers which is selected by mux 122 to address Dowling's data memory 120. As claimed in claim 18, with the "address translation parameter control register located in the address translation memory" and the "address translation unit located in the address translation memory", a read operation may be initiated "to read a first data element at a first address during a read operation", "translating the first address to the second address", and "completing the read operation by reading the second data element a the second address" as claimed in claim 18. To sum up, Dowling does not teach and does not make obvious the elements of claim 18.

Claims 7-10

Claims 7-10 were rejected under 35 U.S.C. §103(a) based on Pechanek 6,173,389 in view of Dowling. The Examiner admits that Pechanek 6,173,389 does not expressly disclose the logic

23

to translate the operand addresses of a sequence of instructions to an RFI sequence of different operand addresses and depends upon Dowling to resolve the admitted deficiency. As noted above, Dowling does not teach and does not make obvious an "instruction register for receiving an instruction encoded with an operand address and control information indicating the operand address is to be translated as part of the instruction's execution" as claimed in claim 7. Thus, Dowling does not resolve the admitted deficiency of Pechanek 6,173,389.

The Examiner suggests that Dowling teaches "using multiplexer to provide addresses not being translated, and to provide translated addresses for subsequent cycles" at page 11, lines 7 and 8 of the Official Action dated 12/26/2007. However, this multiplexer that Examiner suggest Dowling teaches is not what is claimed. Claim 7 claims "receiving a sequence of operand addresses from the multiplexer, and in response to the sequence of RFI operand addresses, translating the sequence of RFI operand addresses to form a sequence of different operand addresses in accordance with the translation pattern". In this manner, the multiplexer of claim 7 makes no selection to provide addresses not being translated or to provide translated addresses. Rather, the multiplexer of claim 7 is used for "selecting between the operand address from the instruction register for a first RFI operation and selecting the RFI update unit's output for subsequent RFI operations". In either case of RFI operations, the operand addresses received from the multiplexer are translated. Dowling does not resolve the deficiencies of Pechanek 6,173,389 in this respect.

Claims 11-14

Claims 11-14 were rejected under 35 U.S.C. §103(a) based on Dowling in view of Choquette. Claim 11 addresses an "address translation memory device for accessing data at translated addresses." The Examiner suggests that Dowling's Fig. 2 processor addresses the

24

address translation memory device having #102 register file, #150 etc. programmable address

translation logic, for translating an instruction #Fig 2, #107 operand address to a different

address and cites col. 13, Table 5 for support.

Dowling's register set 102 of address registers AR0, AR1, ..., ARn, the address

arithmetic unit 106, the programmable address arithmetic unit 212, and the multiplexers 203 and

122 are all part of a function of Dowling's processor to execute instructions, such as load and

store instructions. For example, Dowling's programmable AAU 212 takes an address value from

a selected address register in register set 102 and manipulates the address value for storage back

into an address register in register set 102. An address register from register set 102 may then be

selected for addressing the data memory 120. Dowling's col. 13 Table 5 lists program code

made up of load and store instructions with operand addresses that specify address registers, such.

as address register 0 (AR0), address register 4 (AR4), and address register 5 (AR5). Each of the

load and store instructions specify an address register from the register set 102 AR0-ARn that are

part of Dowling's processor. Dowling operates to manipulate the contents of an identified

address register, as noted above with regard to Figs. 3A and 3C, separately from the data

memory 120. In addition to Dowling's register set 102 of address registers AR0, AR1, ..., ARn,

the address arithmetic unit 106, the programmable address arithmetic unit 212, and the

multiplexers 203 and 122 are also separate from the data memory 120 in order to carry out the

execution of a processor instruction.

Claim 11 addresses, a "storage device located in the address translation memory device

having data accessible at addressable locations, a second read address input internal to the

address translation memory device for selecting data from the storage device during read

operations, and a data output port". The Examiner admits that Dowling does not expressly teach

25

the "claim's aspect of ports associating with the storage device" as claimed by claim 11 and relies

upon Choquette in an effort to resolve this admitted deficiency. Choquette, however, provides

only ports external to the storage device as described in Choquette's paragraph 4. Thus,

Choquette does not resolve the admitted deficiency of Dowling and this claim is not obvious.

An example of using an address translation memory device as claimed serves to illustrate

the contrast between the invention of claim 11 and the combination of Dowling and Choquette.

An exemplary "address translation memory device", the storage unit 810 of Fig. 8A, is described

at page 22, lines 4-10 and lines 15-19 of the present specification. The storage unit 810

comprises a first read address input port with address inputs 815. The storage unit 810 also

comprises a storage device 835 having a second read address input port to the storage device

835. The storage unit 810 further comprises an address translator 830. The combination of the

storage device 835 and address translator 830 located in the address translation memory device

as claimed provides new capability not available in the combination of Dowling and Choquette.

One example is the block move operation described at page 23, line 15 – page 24, line 5. In

order to move a block of four 32-bit registers in a processor without a swap instruction a "total of

twelve 32-bit read operations and twelve 32-bit write operations would be required and would

take at least 12-cycles to accomplish. ...Using the techniques of the present invention, the whole

operation can be accomplished in a single cycle and with no movement of the data ... thus

demonstrating the effectiveness of the present techniques for low power and high performance".

Such a capability is highly advantageous and cannot be accomplished by Dowling in

combination with Choquette.

Claim 15

Claim 15 was rejected under 35 U.S.C. § 103(a) as unpatentable over Dowling,

26

Choquette as applied to claim 14 and further in view of Nair as discussed in the rejection of claim 3. Regarding claim 14, the Examiner did not apply Choquette in the rejection of claim 14. Rather, the Examiner rejected claim 14 based on the same rationale as of claim 2. As noted above, Intel, Choquette, and Nair do not resolve the admitted deficiencies of Dowling and the suggested rationale for rejecting claim 3 is not valid.

B.   The Examiner's Findings of Obviousness are

Also Contrary to Law of the Federal Circuit

    As shown above, the invention claimed is not suggested by the relied upon prior art. The references cited by the Examiner, if anything, teach away from the present invention. It is only in hindsight, after seeing the claimed invention, that the Examiner could combine the references as the Examiner has done. This approach is improper under the law of the Federal Circuit, which has stated that "[w]hen prior art references require selective combination by the Court to render obvious a subsequent invention, there must be some reason for the combination other than the hindsight gleaned from the invention itself." Uniroyal, Inc. v. Rudkin-Wiley Corp., 837 F.2d 1044, 1051, 5 U.S.P.Q. 2d 1434, 1438 (Fed. Cir. 1988), cert. den., 109 S. Ct. 75, 102 L.Ed. 2d 51 (1988); quoting Interconnect Planning Corp. v. Feil, 774 F.2d 1132, 1132, 227 U.S.P.Q. 543, 535 (Fed. Cir. 1985). Furthermore, "[i]t is impermissible to use the claims as a frame and the prior art references as a mosaic to piece together a facsimile of the claimed invention." Uniroyal, 837 F.2d at 1051, 5 U.S.P.Q. 2d at 1438.

    In addition, the Examiner does not appear to have considered "where the references diverge and teach away from the claimed invention", Akzo N.V. v. International Trade Commission, 808 F.2d 1471, 1481, 1 U.S.P.Q. 2d 1241, 1246 (Fed. Cir. 1986), cert. den., 107 S. Ct. 2490, 482 U.S. 909, 107 S.Ct. 2490 (1987); and W.L. Gore Associates, Inc., 721 F.2d 1540,
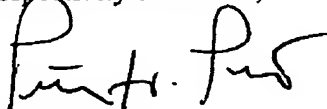
27

220 U.S.P.Q. 303 (Fed. Cir. 1983); nor has the Examiner read the claims as a whole, as required

by statute. 35 U.S.C. §103. See also, <u>Smithkline Diagnostics Inc. v. Helena Laboratories Corp.</u>,

859 F.2d 878, 885, 8 U.S.P.Q. 2d 1468, 1475 (Fed. Cir. 1988); and <u>Interconnect Planning Corp.</u>,

774 F.2d at 1143, 227 U.S.P.Q. at 551.

The Examiner's rejection suggests that the Examiner did not consider and appreciate the

claims as a whole. The claims disclose a unique combination with many features and advantages

not shown in the art. It appears that the Examiner has oversimplified the claims and then

searched the prior art for the constituent parts. Even with the claims as a guide, however, the

Examiner did not recreate the claimed invention.

8.   <u>Conclusion</u>

The rejection of claims 1-19 should be reversed and the application promptly allowed.

Respectfully submitted,

Peter H. Priest
Reg. No. 30,210
Priest & Goldstein, PLLC
5015 Southpark Drive, Suite 230
Durham, NC 27713
(919) 806-1600

28

CLAIMS APPENDIX
(Claims Under Appeal)

1.   A processor address translation apparatus for translating an

instruction operand address to a different operand address, the processor address translation

apparatus comprising:

a memory with an address input for selecting a data element from a plurality of data

elements;

an instruction register for receiving an instruction encoded with an operand address in an

operand address bit field of the instruction and control information indicating the operand

address is to be translated as part of the instruction's execution; and

an address translation unit for accessing the memory in a translation pattern, having the

operand address bit field as input and, in response to the instruction received in the instruction

register, directly translating the operand address bit field received as input to form the different

operand address in accordance with the translation pattern, the different operand address

accessing a data element from the memory through the address input.

2.   The processor address translation apparatus of claim 1 wherein the address

translation unit further comprises:

a plurality of translation parameters and address translation functions supporting a

plurality of translation patterns; and

an input to select a translation pattern from the plurality of supported translation patterns.

3.   The processor address translation apparatus of claim 2 wherein the translation

parameters include k by k s bits and k e bits for a k bit address and address translation functions

further comprises combinatorial logic governed by the following equations, where the operand

29

address bit field input is A0, A1, ..., A(k-1), product operations are treated as ANDs, sum

operations are treated as XORs, and the different operand address is A0', A1', ..., A(k-1)',

$$
\begin{pmatrix} A0' \\ A1' \\ \vdots \\ A(k-1)' \end{pmatrix} = \begin{pmatrix} s0 & s1 & \cdots & s(k-1) & e0 \\ sk & s(k+1) & \cdots & s(2k-1) & e1 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ s(k-1)k & s(k-1)k+1 & \cdots & s(k-1)(k+1) & e(k-1) \end{pmatrix} \times \begin{pmatrix} A0 \\ A1 \\ \vdots \\ A(k-1) \\ 1 \end{pmatrix}.
$$

4.　　　The processor address translation apparatus of claim 1 wherein the instruction is

a block load instruction.

5.　　　The processor address translation apparatus of claim 1 disposed within a plurality

of instruction operand address paths for a plurality of instructions, the plurality of instructions

fetched for simultaneous execution.

6.　　　The processor address translation apparatus of claim 5 wherein the plurality of

instructions constitute a very long instruction word (VLIW).

7.　　　A processor register file indexing (RFI) address translation apparatus for

translating an RFI sequence of instruction operand addresses to an RFI sequence of different

operand addresses, the processor RFI address translation apparatus comprising:

a memory with an address input for selecting a data element from a plurality of data

elements;

an instruction register for receiving an instruction encoded with an operand address and

control information indicating the operand address is to be translated as part of the instruction's

execution;

an RFI update unit enabled to generate on the RFI update unit's output a linear sequence

of RFI operand addresses in response to a received sequence of RFI translation type instructions;

30

a multiplexer for selecting between the operand address from the instruction register for a

first RFI operation and selecting the RFI update unit's output for subsequent RFI operations; and

an address translation unit for accessing the memory in a translation pattern, receiving a

sequence of operand addresses from the multiplexer and, in response to the sequence of RFI

operand addresses, translating the sequence of RFI operand addresses to form a sequence of

different operand addresses in accordance with the translation pattern, the different operand

addresses each accessing a data element from the memory through the address input.

8.   The processor RFI address translation apparatus of claim 7 disposed within PEs of

an array of PEs.

9.   The processor RFI address translation apparatus of claim 7 disposed within a

plurality of instruction operand address paths for a plurality of instructions, the plurality of

instructions fetched for simultaneous execution.

10.   The processor RFI address translation apparatus of claim 9 wherein the plurality

of instructions constitute a very long instruction word (VLIW).

11.   An address translation memory device for accessing data at translated addresses,

the address translation memory device comprising:

a first read address input port to the address translation memory device;

a storage device located in the address translation memory device having data accessible

at addressable locations, a second read address input port internal to the address translation

memory device for selecting data from the storage device during read operations, and a data

output port; and

an address translation unit located in the address translation memory device for accessing

the storage device in a translation pattern, the address translation unit translating a first read

31

address value coupled to the first read address input port in accordance with the translation

pattern, to a second read address value coupled to the storage device second read address input

port for reading data from the storage device at a translated address during a read operation.

12.   The address translation memory device of claim 11 further comprises:

a first write address input port to the address translation memory device;

a storage device located in the address translation memory device having data accessible

at addressable locations, a second write address input port for selecting data in the storage device

during write operations, and a data input port; and

an address translation unit located in the address translation memory device, for

accessing the storage device in a translation pattern, the address translation unit translating a first

write address value coupled to the first write address input port in accordance with the translation

pattern, to a second write address value coupled to the storage device second write address input

port for writing data to the storage device at a translated address during a write operation.

13.   The address translation memory device of claim 11 wherein the storage device

further comprises location selection logic merged with the address translation unit.

14:    The address translation memory device of claim 11 further comprises:

a plurality of translation parameters and address translation functions supporting a

plurality of translation patterns; and

an input to select a translation pattern from the plurality of supported translation patterns.

15.   The address translation memory device of claim 14 wherein the translation

parameters include k by k s bits and k e bits for a k bit address and address translation functions

further comprises combinatorial logic governed by the following equations, where the first read

32

address input is A0, A1, ..., A(k-1), product operations are treated as ANDs, sum operations are

treated as XORs, and translated address output are A0', A1', ..., A(k-1)',

$$
\begin{pmatrix} A0' \\ A1' \\ \vdots \\ A(k-1)' \end{pmatrix} = \begin{pmatrix} s0 & s1 & \cdots & s(k-1) & e0 \\ sk & s(k+1) & \cdots & s(2k-1) & e1 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ s(k-1)k & s(k-1)k+1 & \cdots & s(k-1)(k+1) & e(k-1) \end{pmatrix} \times \begin{pmatrix} A0 \\ A1 \\ \vdots \\ A(k-1) \\ 1 \end{pmatrix}.
$$

16.   A processor address translation method for translating an instruction operand

address to a different operand address, the processor address translation method comprising:

receiving an instruction encoded with an operand address in an operand address bit field

of the instruction and control information indicating the operand address is to be translated as

part of the instruction's execution;

translating directly the operand address bit field received as input according to a function;

and

accessing a data element with the translated address, and repeating the receiving,

translating, and accessing steps to access data elements in a pattern according to the function.

17.   The processor address translation method of claim 16 wherein the function

comprises combinatorial logic for translating the operand address.

18.   An address translation method for translating a first address of a first data element

in an address translation memory to a second address of a second data element in the address

translation memory, the address translation method comprising:

determining a set of {s, e} bits that specify a translation pattern;

loading the set of {s, e} bits into an address translation parameter control register located

in the address translation memory;

33

enabling an address translation unit located in the address translation memory for translation;

initiating a read operation to read a first data element at a first address during a read operation;

translating the first address to the second address in accordance with the {s, e} bit specified translation pattern; and

completing the read operation by reading the second data element at the second address.

19.   The address translation method of claim 16 wherein the set of {s, e} bits include k by k s bits and k e bits for a k bit address and address translation functions further comprises combinatorial logic governed by the following equations, where the operand address bit field input is A0, A1, ..., A(k-1), product operations are treated as ANDs, sum operations are treated as XORs, and the translated address is A0', A1', ..., A(k-1)',

$$\begin{pmatrix} A0' \\ A1' \\ \vdots \\ A(k-1)' \end{pmatrix} = \begin{pmatrix} s0 & s1 & \cdots & s(k-1) & e0 \\ sk & s(k+1) & \cdots & s(2k-1) & e1 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ s(k-1)k & s(k-1)k+1 & \cdots & s(k-1)(k+1) & e(k-1) \end{pmatrix} \times \begin{pmatrix} A0 \\ A1 \\ \vdots \\ A(k-1) \\ 1 \end{pmatrix}.$$

34

EVIDENCE APPENDIX

None.

35

RELATED PROCEEDINGS APPENDIX

None.

36